Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 31 No. 2s (2024)

Analysis of Machine Learning Approaches for DNA Sequencing and Classification: An optimized Approach

Bhushan Bawankar[1], Kotadi Chinnaiah[2], Rajesh Dharmik[3]

[1]Computer Science and Engineering, GHRU, Amravati, India

[2]Computer Science and Engineering, GHRCE, Nagpur, India

[3]Information Technology, YCCE, Nagpur, India

[1]bubawankar@gmail.com; [2]kotadi.chinnaiah@raisoni.net; [3]raj_dharmik@yahoo.com

Article History:

Received: *22-03-2024*

Revised: *10-05-2024*

Accepted: *20-05-2024*

Abstract:

DNA sequencing is essential to contemporary research. It facilitates the advancement of many fields, including phylogenetics, genetics, and meta-genetics. DNA strands must be extracted and read in order to perform DNA sequencing. In order to improve prediction for DNA research and obtain the most accurate results, this research paper compares DNA sequencing using machine learning algorithms. It also aims to efficiently classify DNA sequences according to their features, improving the efficiency and accuracy of DNA sequence classification. The efficiency of various methods is evaluated and contrasted in the study using important metrics like as F1-score, accuracy, precision, and recall. The results indicated that for the human DNA sequence, the Random Forest approach provided the best accuracy of 0.9292 and F1-score of 0.930, while the Genetic algorithm produced higher accuracy of 0.91 and F1-score of 0.913. The potential advancement of genomics research, customized medicine, and various scientific applications that rely on precise classification of DNA sequences is presented by the combination of machine learning and DNA sequencing.

Keywords: Feature extraction, Machine learning, DNA sequences.

1. INTRODUCTION

Today's world is an era of data, with everything in our lives being digitally recorded and everything in our surroundings being connected to a data source. The data may be unstructured, semistructured, or structured. By drawing conclusions from these data, numerous intelligent applications in pertinent fields can be constructed. Therefore, real-world applications are dependent on data management tools and techniques that can quickly and intelligently extract insights or meaningful knowledge from data. These tools and techniques are critically needed [1].

The genetic material of living organisms is found in Deoxyribonucleic Acid (DNA), a lengthy repeating chain of nucleic acids. DNA is the blueprint for life and is essential for reproduction because it transmits genetic information from parent to child. The primary purpose of DNA i.e. as a biological macromolecule, is information storage. DNA sequencing is the process of determining the precise base-pair sequences of adenine, thymine, cytosine, and guanine in a DNA molecule. Information carried by DNA is stored in the form of gene sequences. The rapid growth of DNA sequence data due to the development of sequencing technologies has propelled the study of DNA sequences into the big data wave [2][3].

The machine learning approach is a method that teaches machines how to handle data more accurately and effectively. We are unable to comprehend the information in the data after viewing it. In this case, we predict the data using machine learning approaches. Machine learning is being used by many sectors to extract pertinent data from accessible datasets. A wide variety of algorithms have been developed to enable computers to learn on their own, with the primary goal of machine learning being the extraction of knowledge from available data [2][3][4][5].

The application of machine learning techniques to DNA sequencing and classification aims to improve our comprehension of genetic data, advance computational analysis techniques, and enable a number of applications in biological and medical research. The following are some main goals:

- To increase the precision and effectiveness of sequence alignment, making it possible to identify genetic variants and compare DNA sequences more precisely.
- To precisely detect and categorize genomic variations from sequencing data, with high sensitivity and specificity, differentiating genuine variants from errors or artifacts.
- To label genetic variations and forecast the functional ramifications they will have, such as how they would affect gene regulation, protein structure, or function.
- To annotate genetic variations and explain their relevance for treatment response, illness susceptibility, and other biological processes.
- To improve phylogenetic analysis and identify genetic signals underpinning species divergence and adaption, as well as to reconstruct evolutionary connections more accurately.
- To evaluate and decipher epigenomic data, revealing chromatin accessibility, histone modification, and DNA methylation patterns connected to gene regulation and cellular development.
- To establish a relationship between genetic differences and drug responses, illness susceptibility, and treatment outcomes. This will help identify potential therapeutic targets, prospects for drug repurposing, and individualized treatment plans.
- To provide scalable, interpretable, and resilient machine learning techniques so that biologists and medical professionals may rely on the models' predictions and successfully apply them to their research and clinical practices.

Machine learning techniques can transform the way we study, interpret, and use genetic data by achieving these goals. This will lead to breakthroughs in the fields of biological understanding, medical diagnostics, and therapeutic treatments.

Our goal is to classify DNA sequences using several machine learning (ML) approaches, including K-nearest neighbour, support vector machine, random forest, decision tree, logistic regression, genetic algorithm, ant colony, grid search, gradient boosting, and hill climbing methods, and to compare them with one another. Three publicly available datasets are used to gauge this study's effectiveness. To evaluate the proposed model's performance in terms of F1-score, accuracy, precision, recall on occurrence of DNA sequences.

Md. Ahsan Habib et.al. [4] investigates the utilization of machine learning techniques for classifying DNA sequences, employing algorithms like Support Vector Machines, Random Forests, and Neural Networks on a specified dataset. The study meticulously describes the dataset's origin and preprocessing steps, evaluating model performance using key metrics such as accuracy, precision, recall, and F1-score. Through a comparative analysis, the strengths and limitations of each algorithm in the context of DNA sequence classification are elucidated. The findings offer potential biological insights, and methodological considerations, encompassing feature extraction and validation processes, are highlighted. Concluding with suggestions for future research, the preprint contributes to the evolving landscape of machine learning applications in genomics.

Osisanwo F.Y. et.al. [5] compares various supervised machine learning algorithms for classification. The authors detail the principles and application of algorithms such as Decision Trees, Support Vector Machines, Random Forests, and others. Using a specified dataset, they establish a framework for comparing algorithmic performance based on metrics like accuracy, precision, recall, and F1-score. The results and comparative analysis unveil the strengths and weaknesses of each algorithm, providing insights into their practical implications. The paper concludes with a summary of key findings and potential avenues for future research in the domain of supervised machine learning algorithms for classification.

Varda Venkata Sai Dilip et. al. [6] compares machine learning (Decision Trees, Random Forest, Naive Bayes) and deep learning (Transfer Learning, CNN) algorithms for DNA sequencing. The objective is to enhance prediction models in DNA research. The chosen models, including Decision Tree, Random Forest, Naive Bayes, CNN, and Transfer Learning, exhibit high accuracy. Naive Bayes achieves 98.00% accuracy in machine learning, while Transfer Learning attains 94.57% accuracy in deep learning. These findings emphasize the efficiency of these models in optimizing DNA research predictions across medical domains.

Ahmed El-Tohamy et. al. [9] presents a novel approach for viral DNA sequence classification by integrating deep learning, possibly neural networks, with a genetic algorithm. The study focuses on the classification of viral DNA sequences, providing insights into the methodology, dataset characteristics, and performance evaluation metrics. The results showcase the effectiveness of the proposed deep learning approach, potentially offering advancements in genomics and virology research. The paper concludes by suggesting future research directions, emphasizing the significance of this integrated approach in viral DNA sequence classification.

Belal A. Hamed et. al. [17] focuses on enhancing classification efficiency through machine learning techniques for pattern matching. The study likely delves into the selection and optimization of specific algorithms, offering insights into the methodology, dataset characteristics, and performance metrics. Results are expected to showcase the effectiveness of the chosen machine learning methods in achieving improved classification efficiency. The paper may conclude with implications for pattern matching applications and potential directions for future research in the domain of machine learning optimization for efficient pattern recognition.

2.Data and Different Learning Types in ML

The availability of data is seen to be essential to building a machine learning model or data-driven real-world systems. Data can be in many different forms, including structured, semi-structured, unstructured, and metadata, which is another type that often includes data about the data. We go through several kinds of data in brief, structured i.e. It is highly ordered, readily accessible, and adheres to a data model with a clearly defined structure and are usually kept in tabular form in relational databases. Unstructured i.e. no predetermined structure or format, which makes it considerably more challenging to collect, handle and analyze, including text and multimedia content mostly. Semi-structured i.e. Although data is not kept in a relational database in the same way as structured data, it does have some organizing characteristics that facilitate analysis. Metadata i.e. data are the materials that can be used to classify, quantify, or even document something in relation to an organization's data attributes. However, metadata gives data users greater context for the pertinent data material by describing it [1].

2.1. Learning Types in ML

Learning is the process of acquiring knowledge; individuals naturally pick up knowledge from their experiences due to their capacity for reason. Conversely, conventional computers rely on algorithms for learning. Based on how they approach the learning process in fig.1, many machine learning algorithms that are provided can be categorized into four groups: supervised, unsupervised, semi-supervised, and reinforcement [2].

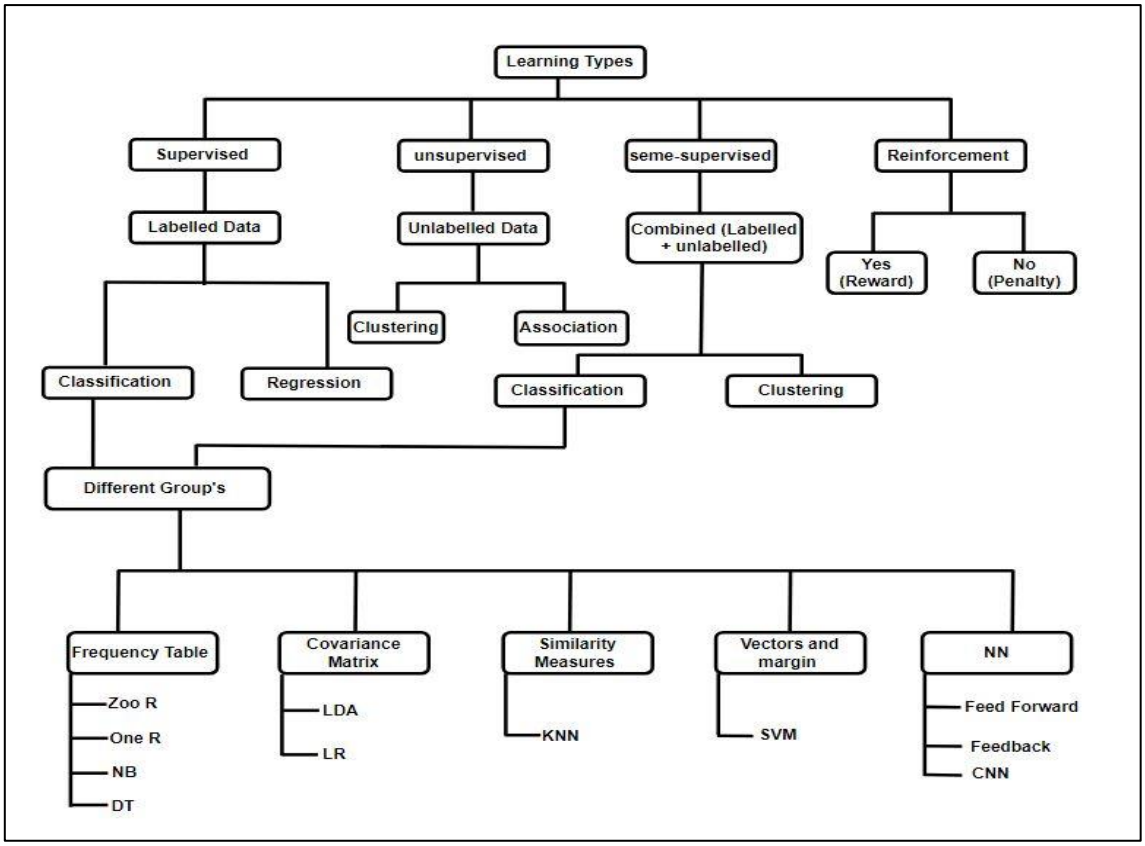


Fig. 1 Different Learning and classifiers based on groups

Supervised- The learning model that efficiently learns how to estimate from training data is constructed through supervised learning. In supervised learning, the complete dataset is split into two portions: the training portion is used to train the classifier, and the remaining portion is used to assess the classifier's accuracy. After it is finished, we can utilize it to test fresh data that will serve as a basis for upcoming information. The two most popular supervised tasks are first regression which fits the data and second classification which divides the data [1][3].

Unsupervised- Unsupervised learning allows for the analysis of unlabelled datasets without requiring human intervention. This is frequently used for exploratory reasons, groupings in findings, generative feature extraction, and the identification of significant patterns and structures. Clustering approach is one of the most popular unsupervised learning [1].

Semi-supervised- Working with both labelled and unlabelled data, semi-supervised learning is sometimes referred to as a hybridization of supervised and unsupervised. In real, semi-supervised learning is helpful since unlabelled data are common and labelled data may be scarce under various circumstances. A semi-supervised learning model's ultimate objective is to produce a better prediction result than one might obtain from the model utilizing just the labelled data [1].

Reinforcement- Reinforcement learning is an environment-driven methodology that makes it possible for machines and software agents to automatically assess the most effective behaviour in a given situation in order to increase productivity. The purpose of this reward-or penalty-based learning approach is to use the knowledge activists provide to take actions that will maximize reward or reduce danger. It's an effective tool that can help automate more jobs or improve the operational efficiency of complex systems like robotics and autonomous driving, among others [1].

Depending on the type of data and the desired result, several kinds of machine learning approaches can be very helpful in building efficient models in different application areas. These techniques can be distinguished by their learning capacities. Below, we present an overview of supervised machine learning techniques that can be used to improve the functionality and intelligence of a data-driven application [1-2].

3. Supervised Machine Learning Algorithms

Researchers' attention has been drawn to machine learning approaches due to technological advancements. In Table 1, shows the comparative analysis of classifiers. Among the supervised machine learning algorithms that focus more on classification are Naïve Bayes Classifier, Support Vector Machine, K-Means Clustering, Random Forest, Decision Tree, and Logistic Regression. Here is a recap of every machine learning approach. [4][5]

3.1 K-Nearest Neighbour

A straightforward supervised machine learning technique, K-Nearest Neighbour categorizes incoming instances according to a similarity metric and may be applied to regression and classification tasks alike. Numerous distance measuring approaches, such as Manhattan, Euclidean, and others, can be

Table 1: Comparative analysis of classifiers

	Decision Tree	Naïve Bayes	SVM	KNN
Accuracy in overall	Avg	Avg	Best	Avg
Speed (Learning + Classification)	Best	Best	Avg	Avg
Tolerance (Missing + irrelevant +redundant) Attributes	Better	Better	Better	Avg
Noise Tolerance	Avg	Better	Avg	Poor
Time Complexity (Upper Bound)	O(L) for Level or depth of tree	O(n*d) for n instances	O(n*d) for n support vectors	O(k*d) for k neighbours

used to detect data similarity. KNN has been utilized in statistical estimate and pattern identification based on their nearest neighbours. Because the KNN algorithm doesn't require training in order to generate predictions, it is faster than other approaches. [3][6]

3.2 Decision Tree

One supervised machine learning technique that can be used for both regression and classification issues is the decision tree.[4] Decision trees are used to create hierarchical categorization models. The process of gradually dividing the dataset into smaller and smaller results in the development of decision trees.[3] There are various nodes. The nodes that store the data in the middle are referred to as internal nodes; the nodes that are at the end are referred to as leaf nodes. The root node is the first node.[6] A choice is represented as a leaf node, and the root node of a tree corresponds to the best predictor found in the datasets provided.[3] The algorithm uses basic decision rules established from training data to develop a training model that can predict the class or value of a target variable. The prediction begins comparing values from the tree's root node and works its way up to the terminal node. [4]

Algorithm:

1. Gini Impurity (for Classification)

The Gini impurity measures the probability of a randomly chosen element being misclassified if it was randomly labeled according to the distribution of labels in the subset.

$$Gini(D) = 1 - \sum(i = 1\ to\ C)(pi^2)$$

- Where pi is the probability of class i in dataset D, and C is the number of classes.

2. Entropy (for Classification)

Entropy measures the impurity or randomness in the data.

$$Entropy(D) = -\sum(i = 1\ to\ C)\big(pi * log2(pi)\big)$$

- Where pi is the probability of class i in dataset D, and C is the number of classes.

3. Information Gain

Information gain measures the reduction in entropy after a dataset is split on a feature.

$$Information\ Gain(D, A) = Entropy(D) - \sum(v\ in\ Values(A))\left(\frac{|Dv|}{|D|}\right) * Entropy(Dv)$$

- Where D is the dataset, A is the feature, Values(A) are the possible values of feature A, Dv is the subset of D where feature A has value v.

4. Variance Reduction (for Regression)

Variance reduction measures the reduction in variance after a dataset is split on a feature.

$$Variance(D) = \left(\frac{1}{|D|}\right) * \sum(i = 1\ to\ |D|)(yi\ -\ \bar{y})^2$$

$$Variance\ Reduction(D, A) = Variance(D) - \sum(v\ in\ Values(A))\left(\frac{|Dv|}{|D|}\right) * Variance(Dv)$$

- Where yi is the target value of the i-th instance, $\bar{y}$ is the mean of target values in D, Dv is the subset of D where feature A has value v.

3.3 Random Forest

Due to adaptability and simplicity, Random Forest is a popular supervised machine learning technique that may be used for both regression and classification issues. [4] As the name suggests, a forest is constructed from numerous trees. The model trains several distinct decision trees on various sets of data, and the average result from these trees is used as the output in the end. [7] The random forest classification algorithm will be used in the same way as the decision tree algorithm.[6] The technique builds numerous decision trees, which are typically trained using the bagging approach, then combines them to provide a more stable and precise evaluation. [4] A straightforward, understandable technique that can handle challenging nonlinear classification tasks is the random forest.[7]

Algorithm:

a. Bootstrap Sampling:

- Generate multiple bootstrap samples from the original dataset. Each sample is created by randomly selecting data points with replacement.
- Let $\{D1, D2, \ldots, DB\}$ be the B bootstrap samples.

b. Build Decision Trees:

- For each bootstrap sample Di, construct a decision tree Ti using a subset of features.
- At each node in the tree, select the best feature from a randomly chosen subset of m features (where m < total number of features).
- Use the chosen feature to split the data and repeat this process recursively to grow the tree.

c. Aggregate Predictions (Ensemble Method):

- For classification:
- Let Ci(x) be the predicted class by the i-th tree for input x.
- The final prediction C^(x) is obtained by majority voting:

$$C\hat{}(x) = mode\{C1(x), C2(x), \ldots, CB(x)\}$$

d. For regression:

- Let Ri(x) be the predicted value by the i-th tree for input x.
- The final prediction R^(x) is obtained by averaging:

$$R^x = \left(\frac{1}{B}\right)\sum(i = 1\ to\ B)Ri(x)$$

e. Out-of-Bag (OOB) Error Estimation:

- For each data point, predict its value using only the trees that did not include this point in their bootstrap sample (OOB prediction).
- Calculate the OOB error by comparing the OOB predictions with the actual values.

f. For classification:

$$OOB\ Error = \left(\frac{1}{N}\right)\sum(i = 1\ to\ N)I\left(C^{OOB(xi)} \neq yi\right)$$

- Where N is the number of data points, C^OOB(xi) is the OOB predicted class for xi, and yi is the true class.

g. For regression:

$$OOB\ Error = \left(\frac{1}{N}\right)\sum(i = 1\ to\ N)\left(R^{OOB(xi)} - yi\right)^2$$

- Where N is the number of data points, R^OOB(xi) is the OOB predicted value for xi, and yi is the true value.

3.4 Naive Bayes

Based on the independence assumptions between predictors and the Bayes theorem, the Naïve Bayes algorithm is a classification method. Large datasets benefit greatly from the simplicity and lack of complex iterative parameter estimation that characterizes Naive Bayesian models.[3][4] With the Algorithm:

1. Calculate Prior Probabilities:

- Compute the prior probability for each class based on the frequency of each class in the training dataset.

$$P(Ck) = (Number\ of\ instances\ in\ class\ Ck)\ /\ (Total\ number\ of\ instances)$$

- Where P(Ck) is the prior probability of class Ck.

2. Calculate Likelihood:

- Calculate the likelihood of each feature given each class. For a feature xi and class Ck, this involves determining the probability of xi occurring in Ck.
- For continuous features, typically assume a Gaussian distribution:

$$P(xi \mid Ck) = (1 / \sqrt{(2\pi\sigma k\text{^}2)}) \exp(-((xi - \mu k)\text{^}2) / (2\sigma k\text{^}2))$$

- Where μk and σk are the mean and standard deviation of feature xi in class Ck.

3. Calculate Posterior Probabilities:

- Use Bayes' theorem to calculate the posterior probability for each class given the feature vector x. The posterior probability is proportional to the product of the prior probability and the likelihood of the observed features:

$$P(Ck \mid x) = (P(Ck) \prod(i = 1 \; to \; n) \; P(xi \mid Ck)) / P(x)$$

- Where P(x) is the evidence (normalizing constant), often not computed directly since it is the same for all classes.

4. Class Prediction:

- Predict the class with the highest posterior probability. This is known as the maximum a posteriori (MAP) decision rule:

$$\hat{C} = arg \; maxCk \; P(Ck \mid x)$$

- Choose the class Ck that maximizes P(Ck | x).

help of this theorem, we may estimate the likelihood that an event will occur and create pairs of probabilities. Compared to other prediction models, a classifier makes predictions easily.[6] Despite its simplicity, the Naive Bayesian classifier is widely used, frequently performs surprisingly well, and outperforms more complex classification techniques.[3]

3.5 Support Vector Machine

One supervised machine learning technology that can be used for regression and classification analysis is the support vector machine.[4] A SVM has a great ability to generalize since it can handle small amounts of data and is less sensitive to noise in a dataset. The hyperplane that maximizes the margin between the two classes is what the SVM seeks to identify. Training non-linear SVM models is possible provided that the right kernel functions are applied. New feature vectors produced by kernel functions typically have more dimensions than the original input. In the new feature space, the SVM locates the new hyperplane, which is linear. [6]

3.6 Logistic Regression

A supervised machine learning technique called logistic regression is typically used to address classification issues, particularly those involving binary classification. The basic operation of this method, which is often referred to as the sigmoid function, is the logistic function. Any actual

number can be entered, and it will convert it to a value between 0 and 1.[4] Similar to a linear regression, a logistic regression builds the curve using the natural logarithm rather than the probability.[3] This approach works well for linearly separable data and is considerably simpler to use, understand, and train. [4]

4. Optimization Algorithms

4.1 Genetic Algorithm

A search-based optimization method founded on the ideas of natural selection and genetics is known as the genetic algorithm. In numerous domains, including scheduling, fuzzy logic control, neural networks, expert systems, optimization design etc. widely used to identify the best answers to challenging issues. Population, assessment, fitness function, selection, crossover, and mutation are some basic terms used in relation to genetic algorithms. The GA codes a chromosome as a solution to a particular issue. After that, it specifies a starting population made up of people who belong to a portion of the problem's solution space. Thus, the solution space in which each workable solution is represented by a unique chromosome is known as the search space. To create the initial population, a random selection of chromosomes is made from the search space prior to the search commencing. Subsequently, the candidates are computationally chosen in a competitive fashion according to their fitness as determined by a certain objective function. The next step is to apply the genetic search operator’s crossover, mutation, and selection one after the other to create a new generation of chromosomes whose predicted quality across the board is higher than that of the previous generation. The best chromosome from the most recent generation is declared as the final solution after this process is repeated until the termination requirement is satisfied. The benefits of using a genetic algorithm include its ability to optimize a wide range of problems, including continuous functions, discrete functions, and multi-objective issues. It also offers a solution that becomes better with time. [8][9]

4.2 Ant Colony

A population-based metaheuristic called "Ant Colony Optimization" can be used to roughly solve challenging optimization issues. Artificial ants, a group of software agents, look for sensible answers given to optimization problems in ACO. The optimization issue is changed to become the optimal path on a weighted graph in order to apply ACO. The artificial ants move on the graph, piece by piece, building solutions. The process of building the solution is random and biased by a pheromone model, which is a collection of parameters related to nodes or edges in the graph, the values of which are changed by the ants during runtime. ACO can be used to identify the best answers to a variety of optimization problems, including the group shop scheduling problem, frequency assignment problem, redundancy allocation problem, traveling salesman problem, and nursing time distribution scheduling. Ant colony benefits include their ability to adapt changes in circumstances, such as new distances, and their efficiency in solving challenges akin to the traveling salesman dilemma [10][11].

4.3 Gradient Boosting

One technique that stands out for its accuracy and speed of prediction, especially when working with big and complicated datasets, is gradient boosting. This algorithm's primary concept is to build models one after the other, with each new model attempting to minimize the mistakes of the

preceding model. To accomplish this, a new model is constructed using the residuals or errors of the old model. Typically, a gradient boosting machine builds an additive model of poorly optimized simple decision trees, which it then generalizes by optimizing an arbitrarily defined loss function to provide more accurate predictions [12][13].

4.4 Grid Search

Grid search is an optimization technique that allows you to choose from a list of parameter alternatives the optimal parameters for your optimization problem. One fundamental tool for hyperparameter optimization is the Grid Search technique. The Grid Search Method selects the hyperparameter combination that yields the lowest error score after taking into account several combinations. A hold-out validation set on the training set is used to evaluate the performance of each combination. The configurations that offer the best performance during the validation process are then produced by the GS algorithm. The grid search's highest hyperparameter values are then used by the model. But when the frequency of the hyperparameters increases, the number of evaluations increases exponentially, and GS becomes useless in the configuration space of high-dimensionality hyperparameters. Therefore, in order to make GS an effective optimization strategy, the hyperparameter setting needs to be limited [14][15].

4.5 Hill Climbing

In order to find the mountain's top, the greedy local search algorithm known as the Hill Climbing algorithm keeps going in the direction of rising value. It just looks at its good local neighbour state and not beyond that, hence it ends when it reaches a peak value when no neighbour has a higher value. The two parts of a node of a hill climbing algorithm are value and state.

When an effective heuristic is available, the hill climbing algorithm is a strategy used to optimize mathematical problems. A well-known illustration of a hill climbing algorithm is the Traveling-Salesman Problem, in which the salesman's journey distance must be minimized [16].

5. Proposed Methodology

In order to investigate the machine learning methods discussed earlier, we decided to construct databases on DNA. The user selects the dataset for design and implementation, after which a few classification and optimization algorithms are run to obtain a comparative study of the methods. To train and deploy the model with the maximum level of accuracy and efficiency, our goal is to determine which optimizer and classifier combination performs best for a given task. The analysis may have employed accuracy, precision, recall, F1-score, and other pertinent performance measures as evaluation metrics. The user has the option to upload or drag and drop the files that they want to compare at the moment of execution. The chosen file needs to be in the *.txt or *.csv extension. Three datasets—human_data.txt, dog.txt, and chimp_data.txt—have been selected for this purpose. Data extraction, DNA sequencing, DNA sequencing utilizing optimization algorithms, optimizing and parallelizing the first two, cross-validation, and hybrid algorithms are among the options available to the user.

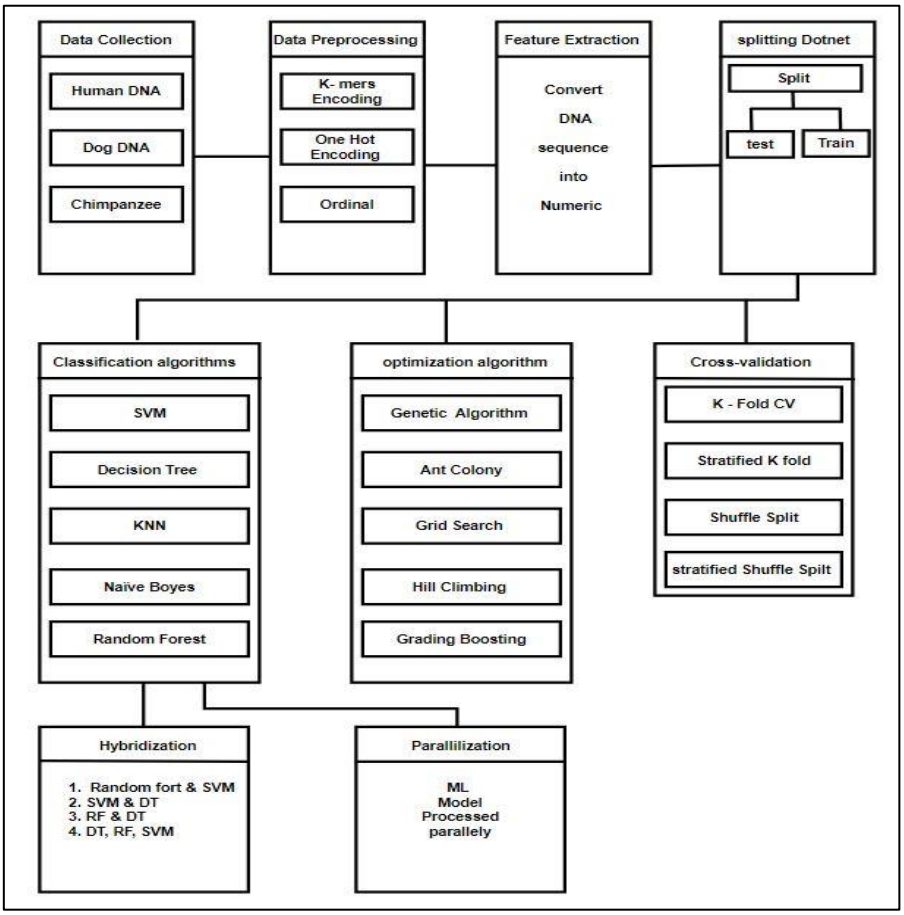


Fig. 3 Proposed Methodology

Figure 3 shows the overall architecture flow and gives a summary of the many stages in our ML technique for DNA sequences. Our approach is divided into multiple stages, each of which contributes differently to the final result. Pre-processing the DNA sequence data, which includes cleaning and filtering the data to eliminate noise and unnecessary information, is the first step in the process. Next, the pre-processed data is put through a process called feature extraction, which entails finding and removing pertinent features from the data. The next step involves building a model that can categorize the DNA sequences according to how similar they are using the attributes that were retrieved. Several machine learning techniques, such as classifiers and optimizers, are used in this paper to create an accurate and effective model [17].

6. Experimental Setup

Fundamentally, our goal is to comprehend the process of employing ML algorithms for DNA sequencing. We are aware that the DNA of humans and other living things is made up of either a distinct sort of sequence or a sequence similar to ATGC. Here, we downloaded a Kaggle data collection containing chimpanzee, dog, and human data. We intend to utilize a classification technique that can effectively classify these specific sequences. We obtained their class and the DNA sequence after processing this data set. Based on sequence, this specific dataset should be able to determine which class a given sequence belongs to. These sequences, which are essentially divided into different types, could represent DNA or gene sequences.

See Figure 4, we used VSCode to develop an experimental Python project and Streamlit to host it on a local web server. Therefore, the user has the option to upload or drag and drop the files that they want to compare at the moment of execution. The chosen file needs to be in the *.txt or *.csv extension. Three datasets—human_data.txt, dog_data.txt, and chimp_data.txt—have been selected for this purpose.

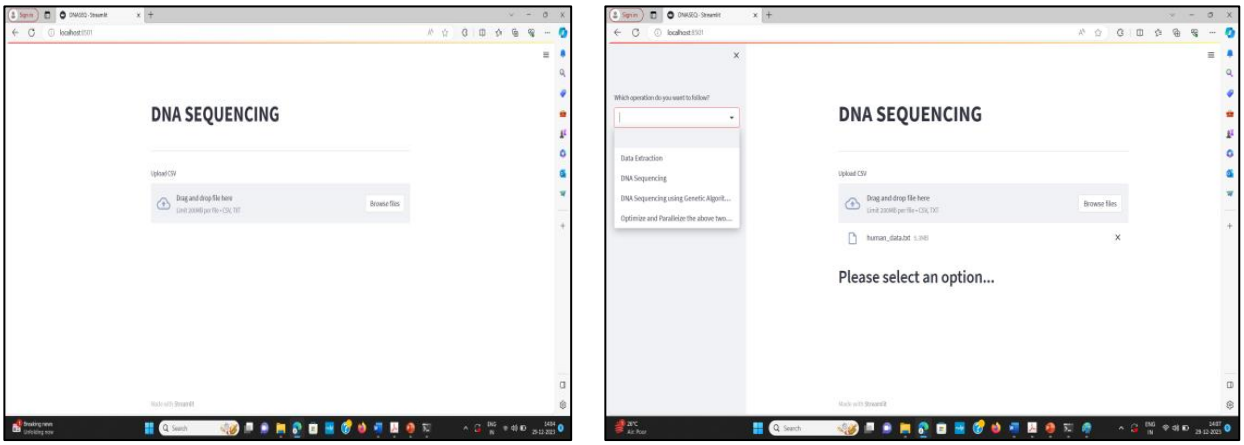


Fig. 4 Experimental setup using Streamlit

We made sure the final dataset was legitimate and that each sample had a matching class label, either 1 or 0. This was done after choosing the dataset and options. A sample of the cleaned dataset is shown in Figure 5, which shows that the data pre-processing step was successfully finished.

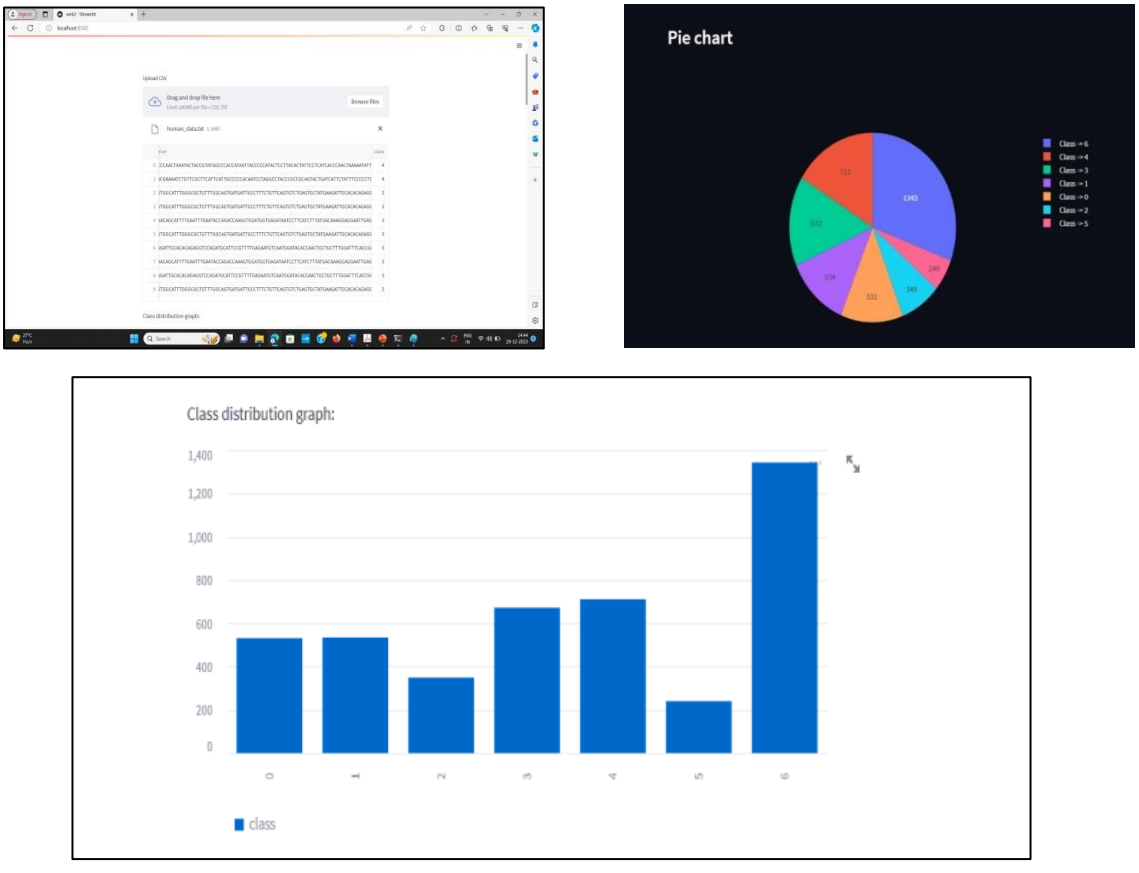


Fig. 5 Class sequence and distribution

Essentially, when working with DNA sequencing, we translate DNA sequences into languages. To do this, we apply k-mer counting, one hot encoding, and ordinal encoding. Here, we take the k-mer counting result into consideration for future implementation. The outcomes of k-mers, one hot, and ordinal encoding is displayed in Figure 6.

The list of k-mers for each gene must then be transformed into string sentences. All of the sequences must now be combined because doing so makes it simple to turn them into a bag of words. Next, using the count vectorizer, we will attempt to transform the strings by applying a bag of words. We accomplished this in order to have our independent feature in the form of strings. Since we are unable to use data key strings straight into our model, we used the count vectorizer to turn the strings into a bag of words. We now verify whether or not the data set is balanced.

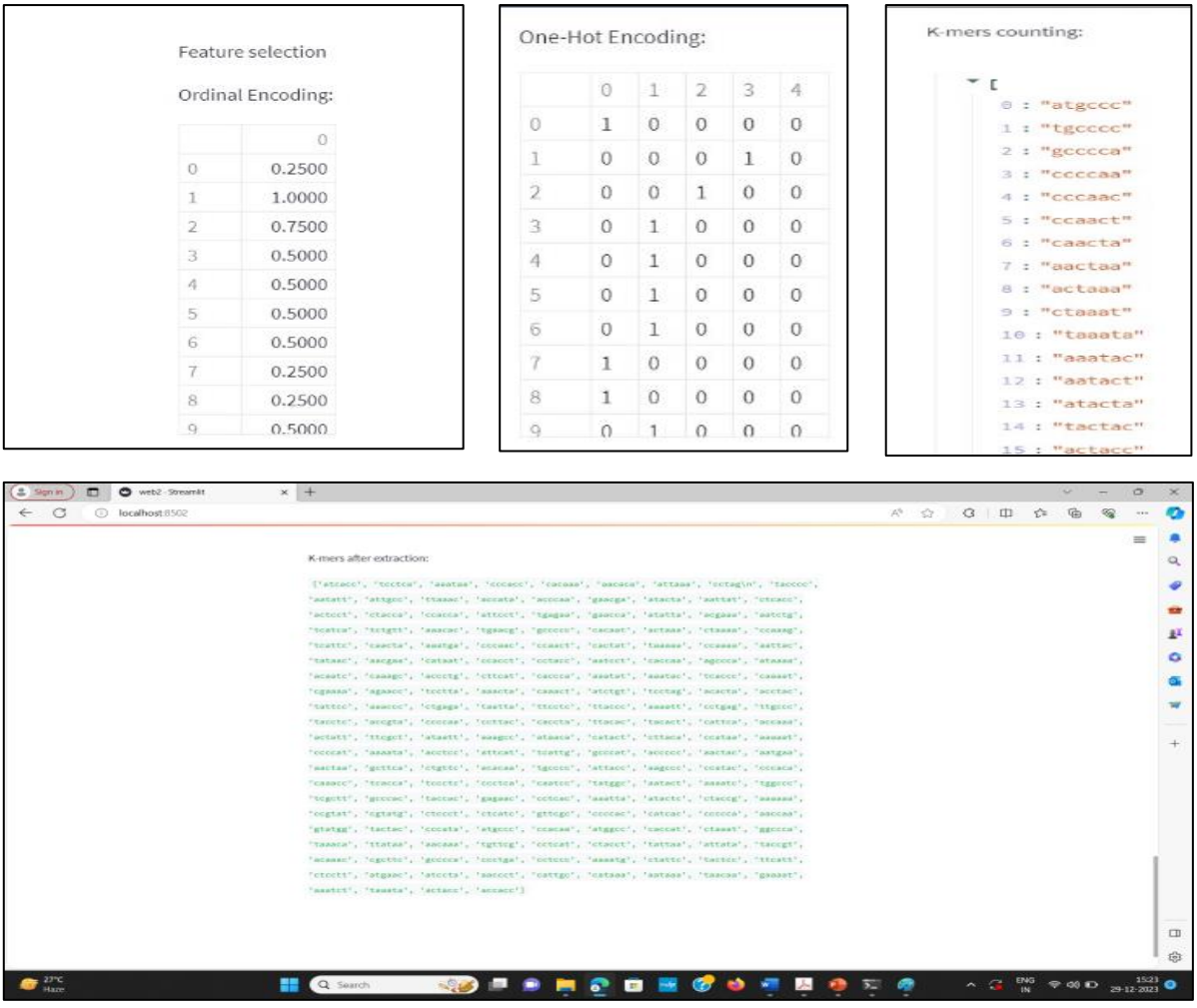


Fig. 6 Feature selection and extraction using ordinal encoding, One-hot encoding and K-mers counting

7. Model Comparison and Analysis

The goal of this paper is to demonstrate the accuracy of the proposed system through a comparison and analysis of various machine learning and optimization techniques. The models that are being proposed are Decision tree, Random Forest, Logistic Regression, SVM, KNN, GA, Ant colony, Hill climbing, Grid search, and Gradient Boosting. The Random Forest method produced better accuracy of 92.92 percent in machine learning as a classifier, and the Genetic algorithm produced better accuracy of 91 percent as an optimizer in human DNA. Table 2, 3 and 4 show the comparison

analysis for three datasets. Similarly, for other datasets, various classifiers and optimizers performed best.

Table 2: Result for Human Dataset

	Algorithm	Accuracy	Precision	Recall	F1
Classifiers	SVM	0.8151	0.882	0.815	0.820
	Random Forest	0.9292	0.937	0.929	0.930
	KNN	0.8607	0.898	0.861	0.863
	Decision Tree	0.8402	0.903	0.840	0.857
	Logistic Regression	0.9258	0.940	0.926	0.927
Optimizers	Genetic Algorithm	0.91	0.930	0.911	0.913
	Ant Colony	0.86	0.744	0.749	0.81
	Gradient Boosting	0.84	0.890	0.873	0.842
	Grid Search	0.90	0.925	0.902	0.904
	Hill Climbing	0.41	0.741	0.405	0.312

Table 3: Result for Dog Dataset

	Algorithm	Accuracy	Precision	Recall	F1
Classifiers	SVM	0.4512	0.822	0.451	0.414
	Random Forest	0.4939	0.833	0.494	0.480
	KNN	0.4390	0.827	0.439	0.403
	Decision Tree	0.5549	0.699	0.555	0.546
	Logistic Regression	0.5183	0.835	0.518	0.510
Optimizers	Genetic Algorithm	0.05	0.833	0.500	0.486
	Ant Colony	0.50	0.758	0.389	0.525
	Gradient Boosting	0.56	0.713	0.561	0.551
	Grid Search	0.51	0.835	0.512	0.501
	Hill Climbing	0.25	0.062	0.250	0.100

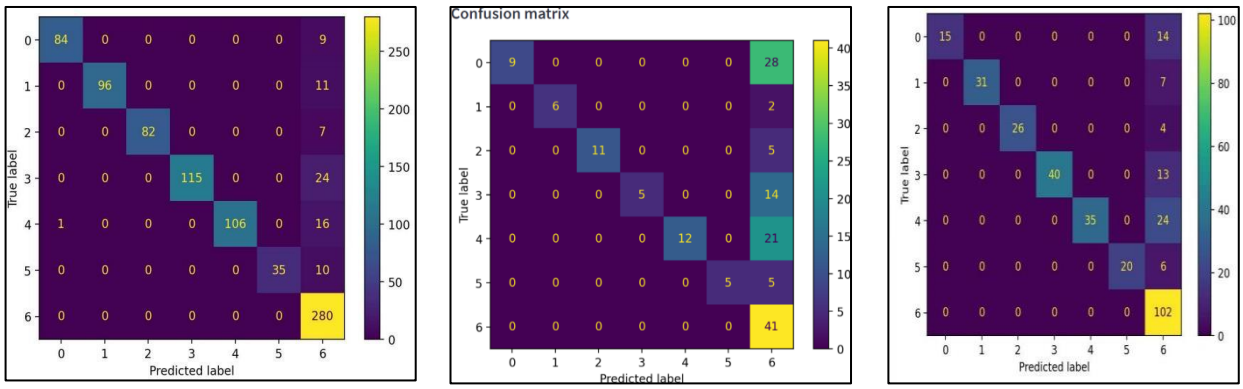
Table 4: Result for Chimpanzee Dataset

	Algorithm	Accuracy	Precision	Recall	F1
Classifiers	SVM	0.7567	0.860	0.757	0.762
	Random Forest	0.7923	0.877	0.792	0.796
	KNN	0.7507	0.904	0.751	0.780
	Decision Tree	0.7834	0.826	0.783	0.782
	Logistic Regression	0.8160	0.886	0.816	0.819
Optimizers	Genetic Algorithm	0.80	0.879	0.798	0.802
	Ant Colony	0.849	0.797	0.738	0.816
	Gradient Boosting	0.78	0.854	0.783	0.787
	Grid Search	0.81	0.883	0.810	0.814
	Hill Climbing	0.37	0.463	0.368	0.256

8. Results and Discussions

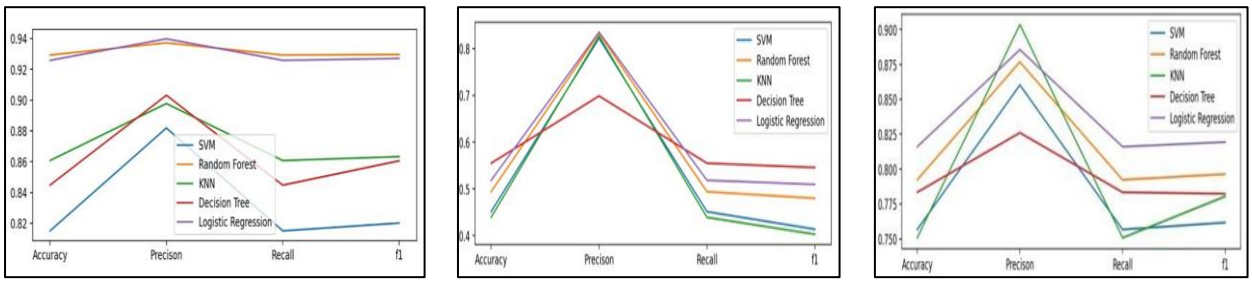
The main goal of this study is to gain a knowledge of how machine learning algorithms can be used to do DNA sequencing. Three datasets, humans, chimpanzees, and dogs, were obtained via Kaggle. Sequences and labels make up each of our datasets, which are split into training and testing ratios of 75% and 25% for machine learning algorithms, respectively. For these datasets, we have performed DNA sequencing and have used various classification methods and optimization strategies. For sequence encoding, we have employed one-hot, k-mer, and ordinal encoding. Here, k-mer encoding has been employed for processing. We transformed the DNA sequences into languages for each machine learning data set by using the k-mer size of six and the k-mer counting technique. Next, we used the count vectorizer to convert the strings by applying a bag of words and the list of k-mers for each gene into string phrases.

Several classification metrics, such as the F1 score, accuracy, recall, and precision, are used to estimate these classification algorithms. The confusion matrix is used to estimate each of the aforementioned characteristics. Figures 7, 8 and 9 below display the confusion matrix and accuracy curve for human, dog, and chimpanzee classifiers and optimizers for DNA sequencing.



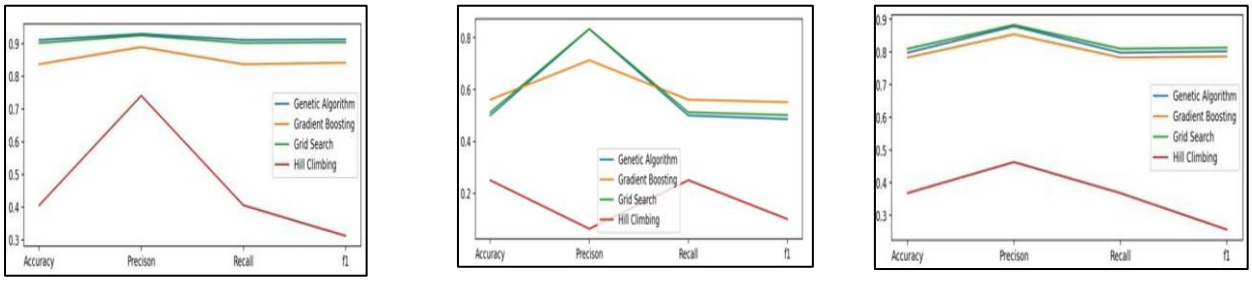
Human DNA Dog DNA Chimpanzee DNA

Fig. 7 Confusion Matrix



Human DNA DOG DNA Chimpanzee DNA

Fig. 8 Accuracy Curve for Classifiers



Human DNA DOG DNA Chimpanzee DNA

Fig. 9 Accuracy Curve for Optimizers

9. Conclusion

The study includes topics like creating and training machine learning algorithms, transforming text inputs to numerical data, and contrasting machine learning methods according to F1-score, accuracy, recall, and precision. In this, all these DNA sequence string encoding namely k-mer counting, One-hot Encoding and ordinal encoding were presented and compared different machine learning algorithms namely Decision tree, Random Forest, SVM, DT, KNN, GA, ACO, Gradient Boosting, Hill Climbing and Grid search with k-mer counting. We found that machine learning algorithms Random Forest, Decision Tree and Logistic regression have performed well with a highest accuracy of 92.92%, 55.49% and 81.60% as classifier and Genetic algorithm, Gradient boosting and Ant Colony have performed well with a highest accuracy of 91%, 56% and 84.9% as optimizer for human, dog and chimpanzee dataset respectively. This dataset is evaluated with different metrics like accuracy, precision, recall and f1-score. In future, the study moves towards the implementation through hybridization, parallelization and cross validation of DNA sequencing with improved accuracy and F-1 score with generalizable optimizer.

References

[1] X. Jiang and C. Xu, “Deep Learning and Machine Learning with Grid Search to Predict Later Occurrence of Breast Cancer Metastasis Using Clinical Data,” J Clin Med, vol. 11, no. 19, Oct. 2022, doi: 10.3390/jcm11195772.

[2] D. A. McCarty, H. W. Kim, and H. K. Lee, “Evaluation of light gradient boosted machine learning technique in large scale land use and land cover classification,” Environments - MDPI, vol. 7, no. 10, pp. 1–22, Oct. 2020, doi: 10.3390/environments7100084.

[3] Y. N. Fuadah, M. A. Pramudito, and K. M. Lim, “An Optimal Approach for Heart Sound Classification Using Grid Search in Hyperparameter Optimization of Machine Learning,” Bioengineering, vol. 10, no. 1, Jan. 2023, doi: 10.3390/bioengineering10010045.

[4] Abdullah-All-Tanvir, I. Ali Khandokar, A. K. M. Muzahidul Islam, S. Islam, and S. Shatabda, “A gradient boosting classifier for purchase intention prediction of online shoppers,” Heliyon, vol. 9, no. 4, Apr. 2023, doi: 10.1016/j.heliyon.2023.e15163.

[5] B. A. Hamed, O. A. S. Ibrahim, and T. Abd El-Hafeez, “Optimizing classification efficiency with machine learning techniques for pattern matching,” J Big Data, vol. 10, no. 1, Dec. 2023, doi: 10.1186/s40537-023-00804-6.

[6] S. Chalup and F. Maire, “A Study On Hill Climbing Algorithms For Neural Network Training.”

[7] C. Blum, M. Y. Vallès, and M. J. Blesa, “An ant colony optimization algorithm for DNA sequencing by hybridization,” Comput Oper Res, vol. 35, no. 11, pp. 3620–3635, Nov. 2008, doi: 10.1016/j.cor.2007.03.007.

[8] W. F. Abd-El-Wahed, A. A. Mousa, and M. A. El-Shorbagy, “Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems,” J Comput Appl Math, vol. 235, no. 5, pp. 1446–1453, Jan. 2011, doi: 10.1016/j.cam.2010.08.030.

[9] I. H. Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions,” SN Computer Science, vol. 2, no. 3. Springer, May 01, 2021. doi: 10.1007/s42979-021-00592-x.

[10] Z. Ibrahim, T. B. Kurniawan, N. K. Khalid, S. Sudin, and M. Khalid, “Implementation of an ant colony system for DNA sequence optimization,” Artif Life Robot, vol. 14, no. 2, pp. 293–296, Nov. 2009, doi: 10.1007/s10015-009-0683-0.

[11] A. El-Tohamy, H. A. Maghwary, and N. Badr, “A Deep Learning Approach for Viral DNA Sequence Classification using Genetic Algorithm.” [Online]. Available: www.ijacsa.thesai.org

[12] X. Zhang, B. Beinke, B. Al Kindhi, and M. Wiering, “Comparing Machine Learning Algorithms with or without Feature Extraction for DNA Classification,” Nov. 2020, [Online]. Available: http://arxiv.org/abs/2011.00485

[13] V. V. S. Dileep*, N. Rishitha, R. Gummadi, and Prof. Natarajan. P, “DNA Sequencing using M achine L earning and D eep L earning A lgorithms,” International Journal of Innovative Technology and Exploring Engineering, vol. 11, no. 10, pp. 20–27, Sep. 2022, doi: 10.35940/ijitee.J9273.09111022.

[14] M. A. Habib, M. Motaleb, and H. Manik, “Classification of DNA Sequence Using Machine Learning Techniques Staircase Detection System View project Natural Language Processing (NLP) View project Classification of DNA Sequence Using Machine Learning Techniques Classification of DNA Sequence Using Machine Learning Techniques”, doi: 10.13140/RG.2.2.31041.63846.

[15] O. F.Y, A. J.E.T, A. O, H. J. O, O. O, and A. J, “Supervised Machine Learning Algorithms: Classification and Comparison,” International Journal of Computer Trends and Technology, vol. 48, no. 3, pp. 128–138, Jun. 2017, doi: 10.14445/22312803/IJCTT-V48P126.

[16] A. S. Alatrany, A. J. Hussain, J. Mustafina, and D. Al-Jumeily, “Machine Learning Approaches and Applications in Genome Wide Association Study for Alzheimer’s Disease: A Systematic Review,” IEEE Access, vol. 10. Institute of Electrical and Electronics Engineers Inc., pp. 62831–62847, 2022. doi: 10.1109/ACCESS.2022.3182543.

[17] R. Vijaya, K. Reddy, and U. Ravi Babu, “A Review on Classification Techniques in Machine Learning.”